CLAIM AMENDMENTS

Claim Amendment Summary

Claims pending

- Before this Amendment: Claims 1-42 and 45-46.

- After this Amendment: Claims 1-42 and 45-46.

Non-Elected, Canceled, or Withdrawn claims: none.

Amended claims: none.

New claims: none.

Claims:

1. (ORIGINAL) A kernel emulator for non-native program modules, the emulator comprising:

an interceptor configured to intercept kernel calls from non-native program modules;

a call-converter configured to convert non-native kernel calls intercepted by the interceptor into native kernel calls.

2. (ORIGINAL) An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate a non-native paradigm for passing parameters into a native paradigm for passing parameters.

3.   (**ORIGINAL**)   An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate non-native CPU instructions into native CPU instructions.

4.   (**ORIGINAL**)   An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate addresses from non-native length into native length.

5.   (**ORIGINAL**)   An emulator as recited in claim 1, wherein the call-converter comprises an argument-converter configured to convert non-native argument format into native argument format.

6.   (**ORIGINAL**)   An emulator as recited in claim 1, wherein the call-converter comprises a translator configured to translate words from non-native word size into native word size.

7.   (**ORIGINAL**)   An emulator as recited in claim 1 further comprising a memory constrainer configured to limit addressable memory to a range addressable by non-native program modules.

8.   (**ORIGINAL**)   An emulator as recited in claim 1 further comprising a shared-memory manager configured to manage memory space that is accessible to both native and non-native program modules.

9. (ORIGINAL)   An emulator as recited in claim 1 further comprising a shared-memory manager configured to synchronize a native shared data structure with a non-native shared data structure.

10. (PREVIOUSLY PRESENTED)   An emulator as recited in claim 1 further comprising a shared-memory manager configured to manage memory space that is accessible to both native and non-native program modules, wherein the shared-memory manager maps versions of process shared data structures (process SDSs) and versions of thread shared data structures (thread SDSs) between native and non-native program modules.

11. (ORIGINAL)   An operating system on a computer-readable medium, comprising:

a native kernel configured to receive calls from native program modules;

a kernel emulator as recited in claim 1 configured to receive calls from non-native program modules.

12. (ORIGINAL)   An operating system on a computer-readable medium, comprising:

a native kernel configured to receive calls from native APIs;

a kernel emulator as recited in claim 1 configured to receive calls from non-native APIs.

421 West Riverside, Suite 500
Spokane, WA 99201
P: 509.324-9256
F: 509.323-8979
www.leehayes.com

lee@hayes

13.   (ORIGINAL)   A method of emulating a kernel for non-native program modules, the method comprising:

   intercepting kernel calls from non-native program modules;

   converting the intercepted non-native kernel calls into native kernel calls.

14.   (ORIGINAL)   A method as recited in claim 13, wherein the converting step comprises translating a non-native paradigm for passing parameters into a native paradigm for passing parameters.

15.   (ORIGINAL)   A method as recited in claim 13, wherein the converting step comprises translating non-native CPU instructions into native CPU instructions.

16.   (ORIGINAL)   A method as recited in claim 13, wherein the converting step comprises translating addresses from non-native length into native length.

17.   (ORIGINAL)   A method as recited in claim 13, wherein the converting step comprises translating words from non-native word size into native word size.

18.   (ORIGINAL)   A method as recited in claim 13 further comprising limiting addressable memory to a range addressable by non-native program modules.

1   **19.   (ORIGINAL)   A** method as recited in claim 13 further

2   comprising synchronizing a native shared data structure with a non-native shared

3   data structure.

4

5   **20.   (ORIGINAL)   A** method as recited in claim 13 further

6   comprising mapping versions of process shared data structures (SDSs) between

7   native and non-native program modules.

8

9   **21.   (ORIGINAL)   A** method as recited in claim 19, wherein a

10   process SDS of a native program module includes a pointer to a process SDS of a

11   non-native program module.

12

13   **22.   (ORIGINAL)   A** method as recited in claim 19, wherein a

14   process SDS of a non-native program module includes a pointer to a process SDS

15   of a native program module.

16

17   **23.   (ORIGINAL)   A** method as recited in claim 13 further

18   comprising mapping versions of thread shared data structures (SDSs) data

19   structure between native and non-native program modules.

20

21   **24.   (ORIGINAL)   A** method as recited in claim 22, wherein a

22   thread SDS of a native program module includes a pointer to a thread SDS of a

23   non-native program module.

24

25

421 West Riverside, Suite 500
Spokane, WA 99201
P: 509.324.9256
F: 509.323.8879
www.leehayes.com

lee@hayes

25. (ORIGINAL)   A method as recited in claim 22, wherein a thread SDS of a non-native program module includes a pointer to a thread SDS of a native program module.

26. (ORIGINAL)   A computer comprising one or more computer-readable media having computer-executable instructions that, when executed by the computer, perform the method as recited in claim 13.

27. (ORIGINAL)   A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 13.

28. (ORIGINAL)   An operating system embodied on a computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 13.

421 West Riverside, Suite 500
Spokane, WA 99201
P: 509.324-9256
F: 509.323-8979
www.leehayes.com

lee@hayes

Serial No.: 09/847,535
Atty Docket No.: MS1-665us   7
RESPONSE TO FINAL OFFICE ACTION DATED
7/13/2005 UNDER 37 C.F.R. § 1.116

1014051524 O:\DOCS\MS\110555US\805551.DOC

atty: Kasey C. Christie

29.   (PREVIOUSLY PRESENTED)   A method comprising:

determining whether an initiating program module is a native or non-native;

if the initiating program is non-native:

limiting available memory to a range that is addressable by the non-native program module, that range of addressable memory being less that the available memory;

establishing non-native a version of a shared memory data structure that may be synchronized with a native version of the same shared memory data structure.

30.   (ORIGINAL)   A method as recited in claim 29 further comprising:

intercepting kernel calls from the non-native program module;

converting the intercepted non-native kernel calls into native kernel calls.

31.   (ORIGINAL)   A method as recited in claim 29 further comprising emulating a non-native kernel for which kernel calls from the non-native program module are intended.

421 West Riverside, Suite 500
Spokane, WA 99201
P: 509.324-9256
F: 509.323-8979
www.leahayes.com

lee@hayes

32. (ORIGINAL) A computer comprising one or more computer-readable media having computer-executable instructions that, when executed by the computer, perform the method as recited in claim 29.

33. (ORIGINAL) A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 29.

34. (ORIGINAL) A method comprising emulating a non-native kernel for a native computing platform so that kernel calls from non-native applications are translated into calls to a native kernel.

35. (ORIGINAL) A method as recited in claim 34, wherein the emulating step comprises:

translating non-native CPU instructions into native CPU instructions;

translating addresses from non-native length into native length;

limiting addressable memory to a range addressable by non-native program modules.

36. (ORIGINAL) A method as recited in claim 35, wherein the emulating step further comprises translating a non-native paradigm for passing parameters into a native paradigm for passing parameters.

37. (ORIGINAL)   A method as recited in claim 34, wherein the converting step further comprises translating words from non-native word size into native word size.

38. (ORIGINAL)   A computer comprising one or more computer-readable media having computer-executable instructions that, when executed by the computer, perform the method as recited in claim 34.

39. (ORIGINAL)   A computer-readable medium having computer-executable instructions that, when executed by a computer, performs the method as recited in claim 34.

40. (ORIGINAL)   A kernel emulator configured to emulate a non-native kernel for a native computing platform so that kernel calls from non-native applications are translated into calls to a native kernel.

41. (ORIGINAL)   An emulator as recited in claim 40, wherein the emulator comprises:

an instruction-translator configured to translate non-native CPU instructions into native CPU instructions;

an address-translator configured to translate addresses from non-native length into native length;

an memory constrainer configured to limit addressable memory to a range addressable by non-native program modules.

42. (PREVIOUSLY PRESENTED)   An operating system on a computer-readable medium, comprising:

a native kernel configured to receive calls from native program modules;

a kernel emulator as recited in claim 40 configured to receive calls from non-native program modules.

43. (CANCELED)

44. (CANCELED)

45. (ORIGINAL)   A kernel emulator for non-native program modules, the emulator comprising:

an interceptor configured to intercept kernel calls from non-native program modules;

a call-converter configured to convert non-native kernel calls intercepted by the interceptor into native kernel calls, wherein the call-converter comprises:

an instruction-translator configured to translate non-native CPU instructions into native CPU instructions;

an address-translator configured to translate addresses from non-native length into native length.

46.   (ORIGINAL)   An operating system on a computer-readable medium, comprising:

a native kernel configured to receive calls from native program modules;

a kernel emulator as recited in claim 45 configured to receive calls from non-native program modules.

Serial No.: 09/847,535
Atty Docket No.: MS1-665us   12   1014051524 O:\DOCS\MS\10665US\805561.DOC
RESPONSE TO FINAL OFFICE ACTION DATED   say: Kasey C. Christie
7/13/2005 UNDER 37 C.F.R. § 1.116

PAGE 16/37 * RCVD AT 11/14/2005 6:55:33 PM [Eastern Standard Time] * SVR:USPTO-EFXRF-6/31 * DNIS:2738300 * CSID:15093238979 * DURATION (mm-ss):09-40